IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of:		
Wei Li, et al.	)	Confirmation No.: 4451
Serial No.: 10/643,628	)	Examiner: Usmaan Saeed
Filed on: August 18, 2003	)	Group Art Unit No.: 2166

For: EXPRESSING FREQUENT ITEMSET COUNTING OPERATIONS

Via EFS-Web
Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

This Appeal Brief is submitted (a) in support of the Notice of Appeal, which was filed on November 11, 2008, and (b) in response to the Notice of Panel Decision from Pre-Appeal Brief Review, which was mailed on December 3, 2008.

I. REAL PARTY IN INTEREST

Oracle International Corporation is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

Appellants are unaware of any related appeals or interferences.

III. STATUS OF CLAIMS

Claims 1-7, 9-20, and 22-30 are pending in this application and were finally rejected in the Final Office Action that was mailed on August 11, 2008. Claims 8 and 21 were canceled during prosecution.

Claims 1-7, 9-20, and 22-30 are the subject of this appeal.

IV. STATUS OF AMENDMENTS

No amendments were filed after the Final Office Action mailed on August 11, 2008.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present application contains independent Claim 1. Claim 1 is generally directed to a method for performing a frequent itemset operation. A database server supports a database language (Specification, paragraph [0007]). Within the database server, a database statement is parsed to detect, within the database statement, a construct that extends the database language (Specification, paragraphs [0028]-[0029]). The construct identifies a function that counts and returns frequent itemsets given a cursor as input to the function (Specification, paragraphs [0031]-[0032] and [0041]). The cursor is used by the function to access values from rows that are returned from a SELECT statement (Specification, paragraphs [0034] and [0046]). The function identifies the frequent itemsets based on the values from the rows returned by the

SELECT statement (Specification, paragraphs [0034] and [0044]-[0046]). The frequent itemset operation is performed as part of execution of the database statement to produce results (Specification, paragraphs [0040]-[0046]). The results are stored in a computer-readable storage medium (Specification, paragraphs [0140] and [0143]-[0144]; FIG. 6, elements 606-610).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-2, 4-7, 12-15, 17-20, and 25-28 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent No. 6,324,533 issued to Agrawal et al. ("*Agrawal*") in view of U.S. Publication No. 2002/0087561 to Chen et al. ("*Chen*").

Claims 3, 9-11, 16, and 22-24 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Agrawal* in view of *Chen*, and further in view of U.S. Patent No. 6,138,117 issued to Bayardo ("*Bayardo*").

Claims 29 and 30 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Agrawal*, in view of *Chen*, in view of *Bayardo*, and further in view of U.S. Patent Publication No. 2002/0059191 to Tamura ("*Tamura*").

VII. ARGUMENTS

It is respectfully submitted that the examiner has erred in rejecting Claims 1-7, 9-20, and 22-30 under 35 U.S.C. § 103(a).

A. CLAIM 1

Claim 1 recites:

> A method for performing a frequent itemset operation, the method comprising the steps of:
> within a database server that supports a particular database language, parsing a database statement to detect within the database statement, a construct that extends the particular language,
> wherein the construct identifies **a function that counts and returns frequent itemsets** given a cursor as input to the function;
> **wherein the cursor is used by the function to access values from rows that are returned from a SELECT statement;**
> **wherein the function identifies said frequent itemsets based on said values from said rows returned by said SELECT statement**;
> performing said frequent itemset operation as part of execution of the database statement to produce results; and
> storing the results in a computer-readable storage medium. (emphasis added)

At least the above-bolded features of Claim 1 are absent from *Agrawal* and *Chen*.

The Background section describes the state of the art that existed at the time the present application was filed:

> Unfortunately, there is a limit to the type of operations that SQL directly supports. Operations that are not directly supported by SQL may be performed by specifying a series of SQL operations which, when executed in combination with each other, perform the desired unsupported operation.
>
> Depending on the nature of the unsupported operation, the combination of SQL operations required to perform the unsupported operation may be quite complex. Further, amount of time and resources required to execute the series of operations may make the use of SQL impractical.
>
> An example of a type of operation that, in general, cannot be performed efficiently using SQL operations is a frequent itemset operation.
>
> When performed using available SQL operations, frequent itemset operations typically require, among other things, **so many join operations** that

> performance is frequently unacceptable when the operation involves any sizable item group population. (emphasis added)

Agrawal describes a frequent itemset counting technique that suffers the same problems identified in the Background of the Invention section. For example, FIGs. 4, 5, 8, 10, and 11 of *Agrawal* depict SQL queries that require numerous join operations and table function calls. Further, col. 6, line 61 to col. 7, line 61 teaches "a process for finding frequent itemsets." That process includes (a) candidate generation, which requires a pruning step, such as is depicted in FIG. 4, and (b) counting support to find frequent itemsets. The counting support is described in detail in col. 8 line 38 to col. 13, line 67 and depicted in FIGs. 8, 10, and 11. Counting support using SQL-92 shows the use of k-way joins, 3-way joins, subquery-based counting, and multiple group-bys. Counting support using SQL with OR extensions also teaches the use of k-way joins.

In contrast, using the invention recited in Claim 1, in order to count and return frequent itemsets, a user merely has to compose a database statement that references a function that:

- **counts and returns frequent itemsets;**
- **has a cursor as an input parameter;**
- **uses the cursor to access values from rows that are returned from a SELECT statement; and**
- **identifies said frequent itemsets based on said values from said rows returned by said SELECT statement.**

Agrawal and *Chen* lack any teaching or suggestion of a **function** that satisfies **any** of these limitations, much less **all** of these limitations.

1. A group-by query cannot be the recited function

The Final Office Action and the Advisory Action equate the group-by-query of *Agrarwal* with the recited function of Claim 1. This is incorrect. A group-by query, as is clear on its face, is **not** a function, but a query that includes a group-by operator. The most analogous

element in Claim 1 to the group-by-query of *Agrawal* is the recited database statement. However, the group-by-query of *Agrawal* does not include a construct that identifies the recited function as Claim 1 requires.

2. *The GatherComb-K table function of* Agrawal *cannot be the recited function*

The Advisory Action equates the GatherComb-K table function of *Agrawal* with the recited function. This is incorrect. The GatherComb-K table function (which is just a merger of the Gather and Comb-K table functions) fails to satisfy **any** of the limitations which are explicitly required of the function recited in Claim 1. Specifically, the GatherComb-K function does not do any of:

- **counting and returning frequent itemsets;**
- **having a cursor as an input parameter;**
- **using the cursor to access values from rows that are returned from a SELECT statement; and**
- **identifying the frequent itemsets based on the values from the rows returned by the SELECT statement.**

The Gather table function of *Agrawal* merely collects all the items of a transaction and outputs a record for the transaction (see col. 10, lines 19-22). Thus, the Gather table function neither counts nor returns frequent itemsets. Therefore, the Gather table function cannot be equated to the recited function of Claim 1.

Also, the Comb-K table function of *Agrawal* cannot be the recited function of Claim 1. According to col. 10, lines 24-27 of *Agrawal*, Comb-K takes as input the output of the Gather table function and merely ***returns* all k-item combinations formed out of the items of a single transaction**. For example, if k = 2 and a transaction contains items A, B, and C, then Comb-K would return {A, B}, {A, C}, and {B, C}. Although Comb-K returns one or more k-item combinations from a transaction, this is far from ***counting* frequent itemsets**, as Claim 1

requires. Further, there is no guarantee that any of the k-item combinations produced by the Comb-K function are frequent. Therefore, the Comb-K table function (as well as the GatherComb-K table function) does not even return frequent itemsets.

Furthermore, from the example database statement (in pseudo-code) in col. 10, lines 41-50 of *Agrawal*, it is clear that Gather and Comb-K are insufficient, by themselves, to count and return frequent itemsets. That example database statement shows that Gather and Comb-K are merely parts of the database statement, the **entirety** of which **must be used** to count and return frequent itemsets. Thus, the GatherComb-K table function does not **count** and **return** frequent itemsets.

Based on the foregoing, *Agrawal* and *Chen* fail to teach or suggest, both individually and in combination, all the features of Claim 1. Therefore, Claim 1 is patentable over *Agrawal* and *Chen*. Withdrawal of the rejection of Claim 1 under 35 U.S.C. § 103(a) is therefore respectfully requested.

3. The combination of Agrawal *and* Chen *is improper*

The Final Office Action asserts that it would have been obvious to combine *Agrawal* and *Chen* because it "would have allowed **Agrawal** to provide high concurrency for the rows in the base table when cursors are used by obtaining a lock on the rows in the base table for the duration of the cursor operation" (page 5; emphasis in Final Office Action). This is incorrect. This rationale merely refers to *Agrawal* and then copies a portion of *Chen*. The copied portion of *Chen* is in no way related to *Agrawal*'s data mining system. The goal of *Chen* is to use cursors more efficiently in order to improve concurrency for a row. However, in *Agrawal*, neither the Gather table function nor the Comb-K table function even takes a cursor as input. Thus, one of ordinary skill in the art would not even think to use in *Agrawal* the modified row-locking cursor approach of *Chen*.

In other words, in order to combine the cursors of *Chen* with *Agrawal*'s data mining system to teach or suggest the recited function of Claim 1, it **must** be alleged that it would have been obvious to use a cursor in the Gather, Comb-K, or GatherComb-K table functions of *Agrawal*. However, it would not have been obvious to use a cursor in either of these table functions. It is unclear how such a combination would even be possible, much less necessary. The Gather table function only takes two simple inputs: a transaction identifier and an item of the transaction associated with the transaction identifier. The Gather table function then outputs a record. There is no reason to modify the Gather table function to take a cursor as input. Similarly, the Comb-K table function only takes two simple inputs: a transaction identifier and a list of items of the transaction associated with the transaction identifier. There is no reason to modify the Comb-K table function to take a cursor as input.

B. CLAIMS 2-7, 9-17, 27, AND 29

Claims 2-7, 9-17, 27, and 29 are dependent claims that depend (indirectly or directly) on Claim 1 discussed above. Therefore, each of Claims 2-7, 9-17, 27, and 29 includes the same features of Claim 1 discussed above. Each of Claims 2-7, 9-17, 27, and 29 is therefore patentable over the cited art for at least the same reasons discussed above for Claim 1. Furthermore, each of Claims 2-7, 9-17, 27, and 29 recite additional limitations that independently renders it patentable. Examples follow.

1. Claim 9

Claim 9 depends on Claim 3 and additionally recites that additional criteria, specified in the database statement, specifies a minimum length. Claim 9 further recites that performing the frequent itemset operation includes performing a frequent itemset operation whose results exclude all itemsets that include fewer items than the minimum length specified by the

additional criteria. For example, if a frequent itemset were {A, B, C}, but the minimum length were four, then {A, B, C} would be excluded from the results.

The Final Office Action cites col. 5, lines 21-23 of *Agrawal* for allegedly disclosing Claim 9. This is incorrect. The only similarity between these quoted portions and Claim 9 is the word "minimum." However, these quoted portions of *Agrawal* are referring to minimum **support**, not minimum **length**. According to *Agrawal*, the "support of an item is the number of **transactions** that contain the item" (col. 2, lines 57-58). In contrast, the "minimum length" of Claim 9 is referring to the number of **items** in an itemset. Therefore, the item set {A, B, C} may have minimum support (e.g., 20+ transactions contain each of those items) but may still be less than the minimum length (e.g., four).

Furthermore, even if *Agrawal* did teach a minimum length (which it doesn't), *Agrawal* and *Bayardo* still fail to teach or suggest (individually and in combination) that the minimum length is specified in the database statement.

2. Claim 10

Claim 10 depends on Claim 3 and additionally recites that additional criteria, specified in the database statement, specifies a maximum length. Claim 9 further recites that performing the frequent itemset operation includes performing a frequent itemset operation whose results exclude all itemsets that include more items than the maximum length specified by the additional criteria. For example, if a frequent itemset were {A, B, C, D}, but the maxium length were three, then {A, B, C, D} would be excluded from the results.

The Final Office Action quotes col. 8, lines 4-6; col. 5, lines 56-60 of *Agrawal* for allegedly disclosing Claim 10, except for the recited "maximum length." The Final Office Action then quotes col. 1, lines 22-26 of *Bayardo* for allegedly disclosing "maximum length." This is incorrect. Those quoted portions merely state:

> F consists of k+2 attributes (item.sub.1, . . . , item.sub.k, support, len), where k is the size of the largest frequent itemset and len is the length of the itemset.
>
> In particular, it is not practical to assume that all items in a transaction appear as different columns of a single tuple because often the number of items per transaction can be more than the maximum number of columns that the database supports. For instance, for one of our real-life datasets the maximum number of items per transaction is 872 and for another it is 700.
>
> For the most part, frequent-pattern mining methods have been developed to operate on databases in which the longest frequent patterns are relatively short, e.g., those with less than 10 items. (emphasis added)

The Final Office Action states, "Examiner interprets the length of 10 as the maximum length." However, each of these quoted portions fail to teach or suggest that the recited maximum length is specified in a database statement. For example, the first quoted portion (from *Agrawal*) merely states that the schema of a table F, which contains the frequent itemsets (see col. 3, lines 11-13 and col. 8, lines 1-4), consists of k items, support, and length, which is the length of the corresponding frequent itemset. However, this length value is not specified in a database statement, nor does this length value limit the any of the frequent itemsets by excluding itemsets that include more items than the length value, as Claim 10 would require.

The second quoted portion (from *Agrawal*) merely states that the highest number of items per transaction has reached 872 in one situation and 700 in another situation. However, the second quoted portion fails to teach or suggest that 872 and 700 are specified in a database statement, much less that such numbers limit the resulting frequent itemsets by excluding itemsets that include more items than those numbers, as Claim 10 would require.

The third quoted portion (from *Bayardo*) merely points to the fact that the longest frequent patterns in databases are relatively short. Again, this quoted portion fails to teach or suggest any sort of maximum length is specified in a database statement, much less that the

number '10' limits the resulting frequent itemsets by excluding itemsets that include more items than '10,' , as Claim 10 would require.

3. Claim 11

Claim 11 depends on Claim 3 and additionally recites that additional criteria, specified in the database statement, specifies a set of one or more included items. Claim 11 further recites that performing the frequent itemset operation includes performing a frequent itemset operation whose results exclude all itemsets that do not include all items in the set of one or more included items. For example, if the set of included items were A and B, then any frequent itemset that did not include A and B would be excluded from the resulting frequent itemsets.

The Final Office Action cites col. 3, lines 2-16 of *Agrawal* for allegedly disclosing Claim 11, except for "one or more included items." By dissecting Claim 11 in this manner, the Final Office Action destroys the meaning of Claim 11. MPEP 2106(II)(c) states that "when evaluating the scope of a claim, every limitation in the claim must be considered. USPTO personnel may not dissect a claimed invention into discrete elements and then evaluate the elements in isolation. Instead, the claim as a whole must be considered." However, the Final Office Action does precisely this, i.e., the Final Office Action dissects Claim 11 by removing the critical language "one or more include items" from Claim 11 and evaluating that phrase in isolation. Without the phrase "one or more included items", it is not clear what meaning Claim 11 would have.

In order to render Claim 11 unpatentable over *Agrawal* and *Bayardo*, the Final Office Action must show that such a combination teaches or suggests (1) that a database statement specifies one or more included items, and (2) if the one or more included items are not included in an itemset, then *Agrawal* or *Bayardo* excludes that itemset from the result of performing the frequent itemset operation. However, both *Agrawal* and *Bayardo* fail to teach or suggest that

one or more included items are specified in a database statement. The Final Office Action cites col. 3, lines 42-45 of *Bayardo*, which states "(1) generating an initial set C of candidates where each candidate c includes two distinct sets of items: c.head and c.tail." Neither this cited portion of *Bayardo* nor the cited portion of *Agrawal* teaches or suggests that one or more included items are specified in a database statement. Indeed, both the cited portion of *Agrawal* and the cited portion of *Bayardo* fail to even refer to a database statement.

Furthermore, the cited portion of *Agrawal* is concerned with generating rules from already determined frequent itemsets. Therefore, the cited portion of *Agrawal* is completely unrelated to what the database statement (i.e., that initiates the counting and returning of frequent itemsets) specifies.

4. Claim 29

Claim 29 is similar to Claim 11 in that it depends on Claim 3 and additionally recites what the database statement specifies. Instead of a set of one or more included items, Claim 29 recites a set of one or more excluded items. To satisfy Claim 29, itemsets that include all the items in this set of excluded items are excluded from the results of performing the frequent itemset operation. For example, if the set of excluded items were A and B, then any frequent itemset that did include A and B would be excluded form the resulting frequent itemsets.

The Final Office Action concedes that *Agrawal*, *Chen*, and *Bayardo* do not disclose Claim 29. The Final Office Action then cites paragraph 21 of *Tamura* for allegedly disclosing Claim 29. This is incorrect. That paragraph of *Tamura* merely states that an itemset that includes a combination that is not in a frequent (k-1)-itemset is excluded from the candidate k-itemset. This paragraph is referring to one stage in the basic algorithm for determining a k-itemset. Again, this portion mentions nothing about items being specified in a database statement. None of the cited references teach such a database statement, which allows database

users to be very flexible in the exact data they want to view.

C. CLAIMS 14-20, 22-26, 28, AND 30

Claims 14-20, 22-26, 28, and 30 are computer-readable storage medium claims that depend on one of the claims discussed above. Each of Claims 14-20, 22-26, 28, and 30 is therefore patentable over the cited art for at least the same reasons discussed above for claim upon which it depends.

D. CONCLUSION AND PRAYER FOR RELIEF

Based on the foregoing, it is respectfully submitted that the rejection of Claims 1-40 under 35 U.S.C. § 103(a) as being unpatentable over the cited art lacks the requisite factual and legal bases. Appellants therefore respectfully request that the Honorable Board reverse the rejection of Claims 1-40 under 35 U.S.C. § 103(a).

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

/DanielDLedesma#57181/

Daniel D. Ledesma
Reg. No. 57,181

Date: January 12, 2009

2055 Gateway Place, Suite 550
San Jose, CA 95110-1083
Telephone: (408) 414-1080 ext. 229
Facsimile: (408) 414-1076

VIII. CLAIMS APPENDIX

1. (previously presented) A method for performing a frequent itemset operation, the method comprising the steps of:

within a database server that supports a particular database language, parsing a database statement to detect within the database statement, a construct that extends the particular language,

wherein the construct identifies a function that counts and returns frequent itemsets given a cursor as input to the function;

wherein the cursor is used by the function to access values from rows that are returned from a SELECT statement;

wherein the function identifies said frequent itemsets based on said values from said rows returned by said SELECT statement;

performing said frequent itemset operation as part of execution of the database statement to produce results; and

storing the results in a computer-readable storage medium.

2. (Previously presented) The method of Claim 1, wherein the database statement is expressed in a particular database language, and wherein the particular database language is SQL.

3. (previously presented) The method of Claim 1, wherein:

the database statement specifies frequency criteria and additional criteria;

said frequency criteria specifies at least one criterion that relates to how frequently combinations of items appear together;

said additional criteria do not specify any criterion that relates to how frequently combinations of items appear together;

the additional criteria specify at least one of (a) a minimum length, (b) a maximum length, (c) a set of one or more included items; or (d) a set of one or more excluded items; and

the results include frequent itemsets that satisfy both said frequency criteria and said additional criteria, and wherein the results do not include frequent itemsets that satisfy said frequency criteria but do not satisfy said additional criteria.

4. (Previously Presented) The method of Claim 1 wherein:

the database statement includes a first indication of a first input format;

the frequent itemset operation operates on input that conforms to said first input format; and

the method further comprises the steps of:

parsing a second database statement to detect within the second database statement a construct that extends a database language, wherein the second database statement includes a second indication of a second input format that is different from said first input format; and

in response to detection of said construct in said second database statement, the database server performing a second frequent itemset operation as part of execution of the second database statement, wherein the second frequent itemset operation operates on input that conforms to said second format.

5. (Original) The method of Claim 4 wherein the first indication is identification of a first table function and the second indication is identification of a second table function.

6. (Original) The method of Claim 1 wherein the frequent itemset operation uses, as input, a row source that is generated during execution of other operations specified in said database statement.

7. (Original) The method of Claim 1 wherein the frequent itemset operation produces, as output, a row source that is used as input for other operations specified in said database statement.

8. (Canceled)

9. (Previously Presented) The method of Claim 3 wherein:

the additional criteria specify a minimum length; and

the step of performing the frequent itemset operation includes performing a frequent itemset operation whose results exclude all item sets that include fewer items than the minimum length specified by the additional criteria.

10. (Previously Presented) The method of Claim 3 wherein:

the additional criteria specify a maximum length; and

the step of performing the frequent itemset operation includes performing a frequent itemset operation whose results exclude all item sets that include more items than the maximum length specified by the additional criteria.

11. (Previously Presented) The method of Claim 3 wherein:

the additional criteria specify a set of one or more included items; and

the step of performing the frequent itemset operation includes performing a frequent itemset operation whose results exclude all itemsets that do not include all items in said set of one or more included items.

12. (Original) The method of Claim 1 wherein the step of performing the frequent itemset operation includes performing a frequent itemset operation whose results identify

frequent itemsets, and

for each of the frequent itemsets, a count of how many item groups included the frequent itemset.

13. (Previously Presented) The method of Claim 1 wherein the step of performing the frequent itemset operation includes performing a frequent itemset operation whose results identify

frequent itemsets, and

for each of the frequent itemsets, a count of how many items are in the frequent itemset.

14. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 1.

15. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 2.

16. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 3.

17. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 4.

18. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 5.

19. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 6.

20. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 7.

21. (Canceled)

22. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 9.

23. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 10.

24. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 11.

25. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 12.

26. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 13.

27. (Previously presented) The method of Claim 1, wherein the construct is a table function.

28. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 27.

29. (Previously presented) The method of Claim 3 wherein:

the additional criteria specify a set of one or more excluded items; and

the step of performing the frequent itemset operation includes performing a frequent itemset operation whose results exclude all itemsets that include all items in said set of one or more excluded items.

30. (Previously presented) A computer-readable storage medium carrying one or more sequences of instructions which, when executed by one or more processors, causes the one or more processors to perform the method recited in Claim 29.

IX. EVIDENCE APPENDIX PAGE

None.

X. RELATED PROCEEDINGS APPENDIX PAGE

None.